2019

CSE2004
Database Management System
Review III

SIDDHARTH GARG (18BCB0038)
THIRUMURUGAN A (18BCE2060)

- SUBMITTED TO – PROFFESOR Geetha Mary

Slot- L55 + L56

Introduction

- Website particularly focuses on food park mess system(student credit based)
- Drawbacks of the system- takes a long time to prepare the food
- To help students save time, the website will be helpful
- Also allows the admins to keep track of orders
- External schema of the website- two views(admin and student)
- Admin login features- modify student and food details
- Student login features- order food and check balance

Abstract

- Topic- Food park Mess management system
- Enables students to order food from the mess online
- Saves students' time
- The project aims to reduce the complexities of a student's daily routine by modernising mess system.

Requirements For Application

Software Requirements:

1. SQLite
2. Django
3. HTML
4. CSS
5. JavaScript
6. Bootstrap
7. A Server
8. Domain Address to host the application
9. Storage Space

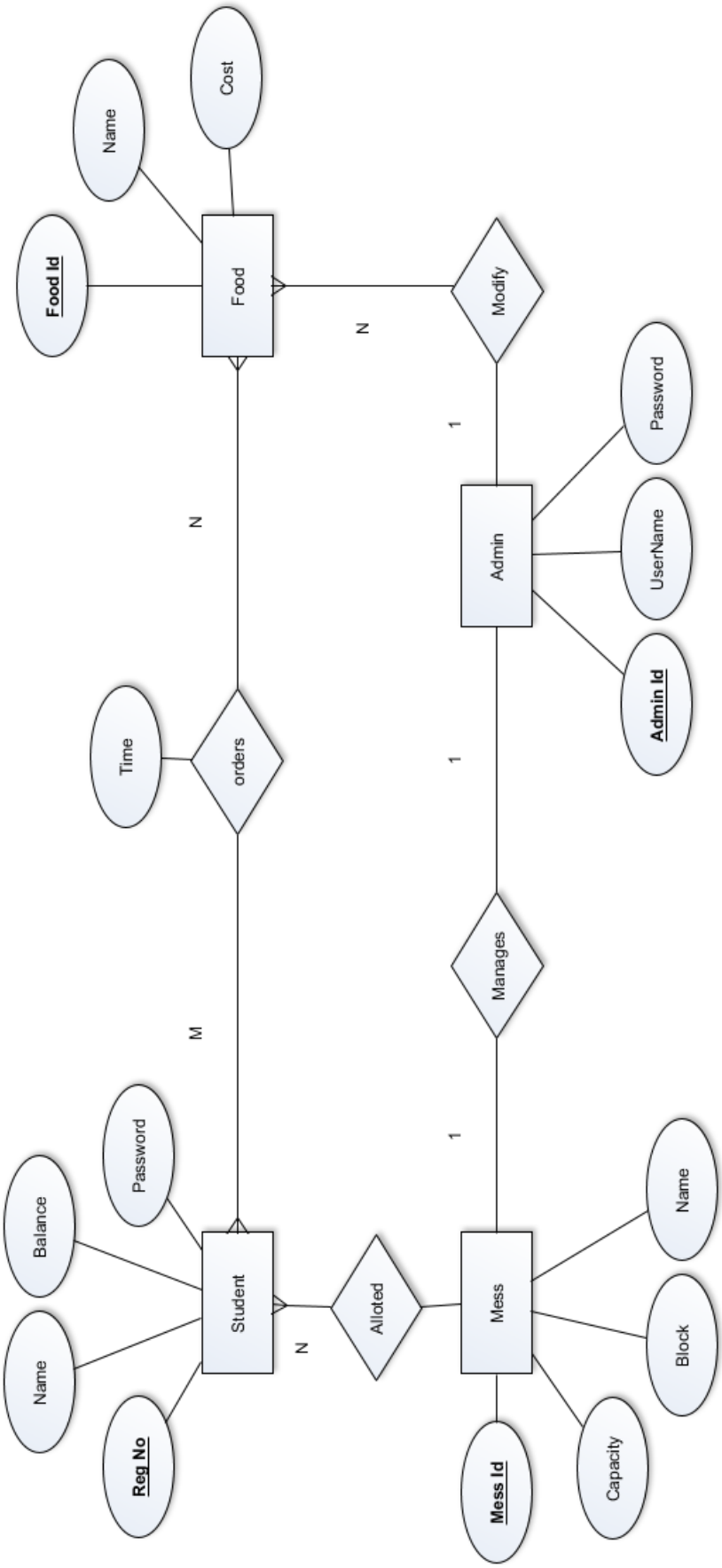
Hardware Requirements:

1. Server Infrastructure
2. Admin Devices
3. User Device (Laptop / Mobile)
4. Storage for database

Database Requirements:

1. Each STUDENTS is allotted a MESS. Many STUDENTS can belong to the same MESS.
2. We store Name, Reg No, Balance and Password as attributes for every STUDENT.
3. We can keep track of current Balance of STUDENT after every order.
4. We store Mess_Id, Capacity, Block and Name as attributes for each MESS.
5. One MESS will contain multiple number of STUDENTS. Approximately 350 STUDENTS will be allotted in one MESS.
6. Each STUDENT is allotted one MESS.
7. Each STUDENT can order FOOD. We keep track of FOOD orders with order_id
8. Every Mess has one ADMIN.
9. One ADMIN can manage only one MESS.
10. We store the Admin_id, Username, Password of every ADMIN.
11. ADMINS can Modify FOOD Costs and other attributes.
12. After every order the balance of STUDENT is updated.
13. An ADMIN can control the attributes of a MESS.

ER Diagram


Cost
Name
Food Id
Food
Modify
N
1
Password
UserName
Admin Id
Admin
N
Time
orders
1
Manages
M
1
Password
Balance
Name
Reg No
Student
N
Alloted
Mess
Name
Block
Capacity
Mess Id

ER Diagram to Relational Mapping:

Student:

Student_id	Name	*Mess_id*	Balance	Password

Mess:

Mess_id	Block	Name	Capacity	Admin

Admin User(mess):

Admin_id	Username	Password

Food:

Food_id	*Admin*	Name	Cost

Order:

Order_id	*Food_id*	Time

Tables:

1. Student:

Attribute	**Data type**	**Constraints**
Student_id	VARCHAR(9)	Primary key
Name	VARCHAR(20)	NOT NULL
Mess_id	VARCHAR(5)	Foreign key of mess
Password	VARCHAR(64)	NOT NULL, LEN()>7
Balance	FLOAT	NOT NULL

2. Mess:

Attribute	**Data type**	**Constraints**
Mess_id	VARCHAR(5)	Primary key
Block	VARCHAR(1)	NOT NULL
Name	VARCHAR(10)	NOT NULL

3. Admin User:

Attribute	**Data type**	**Constraints**
Admin_id	INT	Primary key
Username	VARCHAR(10)	NOT NULL
Password	VARCHAR(64)	NOT NULL, LEN()>7

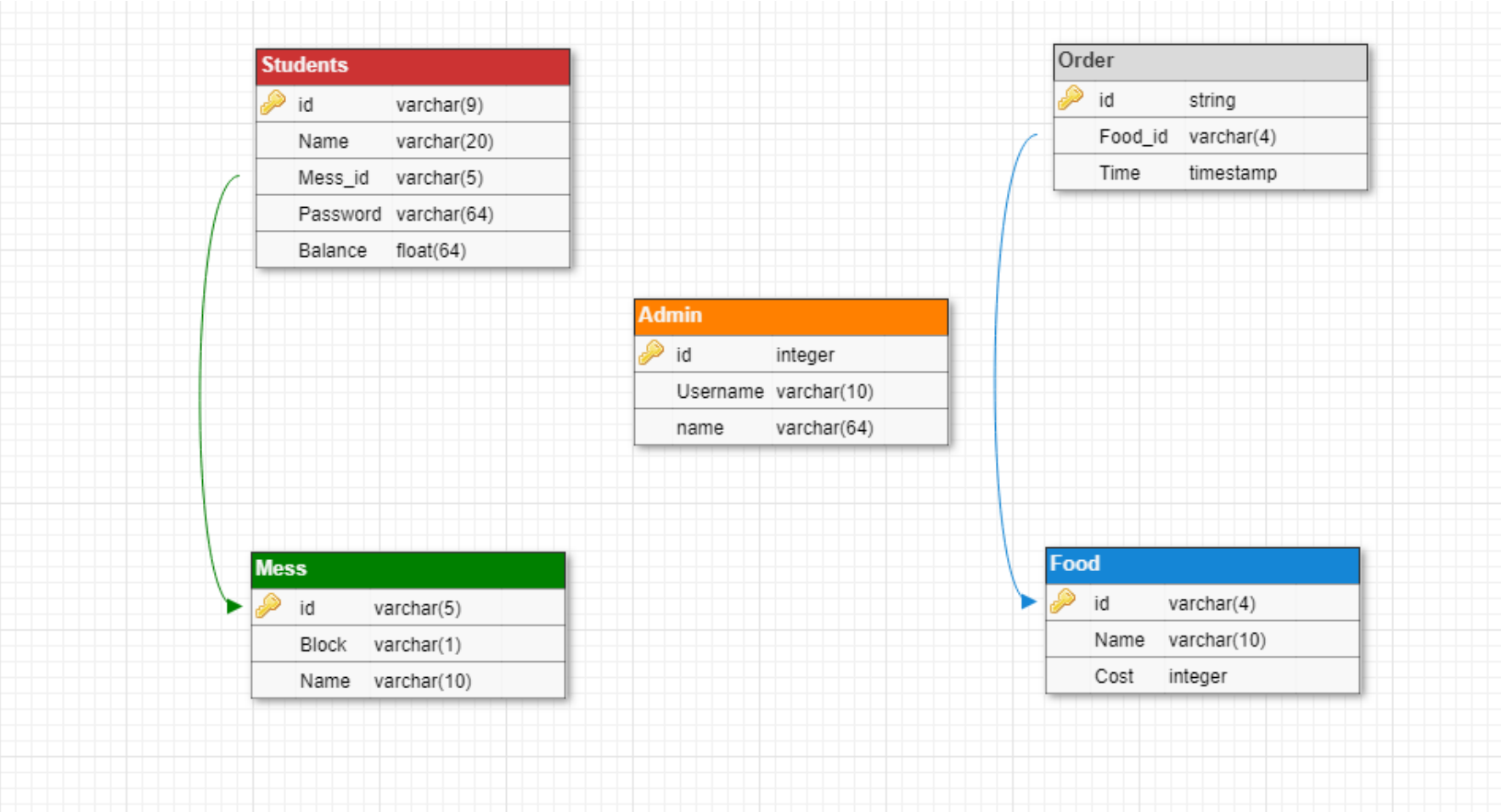
4. Food:

Attribute	**Data type**	**Constraints**
Food_id	VARCHAR(4)	Primary key
Name	VARCHAR(10)	NOT NULL
Cost	INT	NOT NULL

5. Order:

Attribute	**Data type**	**Constraints**
Order_id	INT	Primary key
Food_id	VARCHAR(4)	Foreign key of food
Time	Date	NOT NULL

Schema Diagram



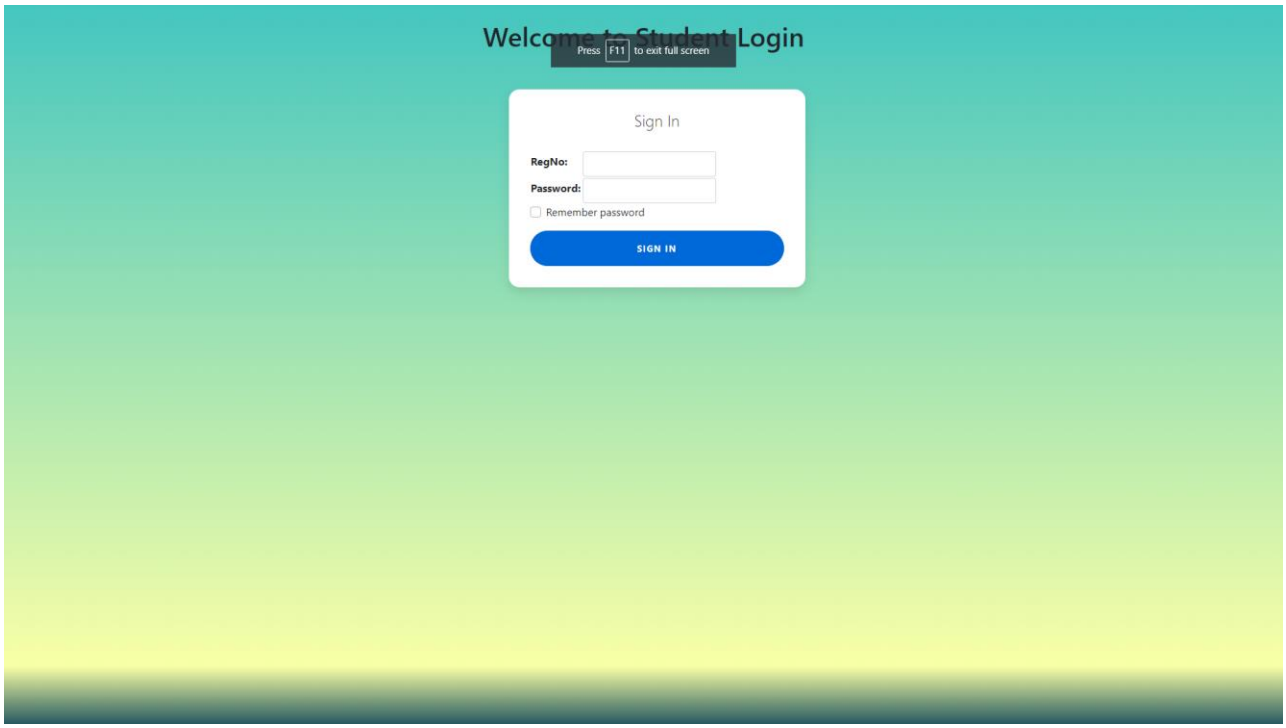
(18BCE2060,18BCB0038)

CSE2004: Review

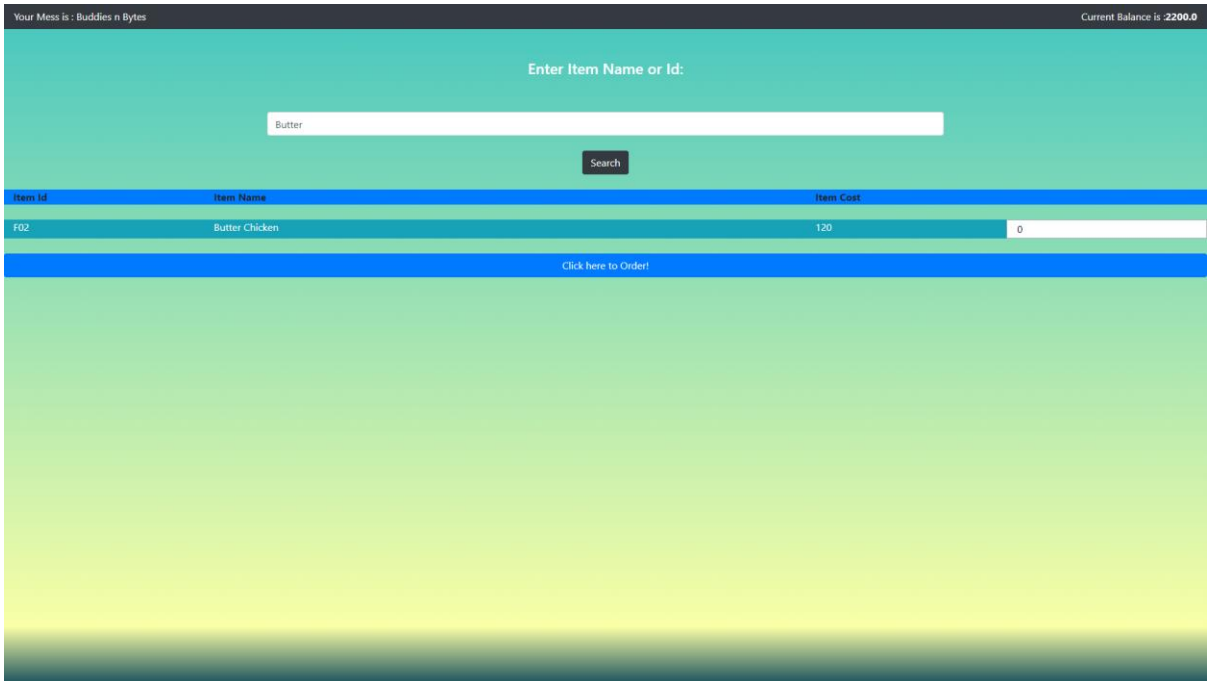
Submitted to **Prof. Geetha Mary**

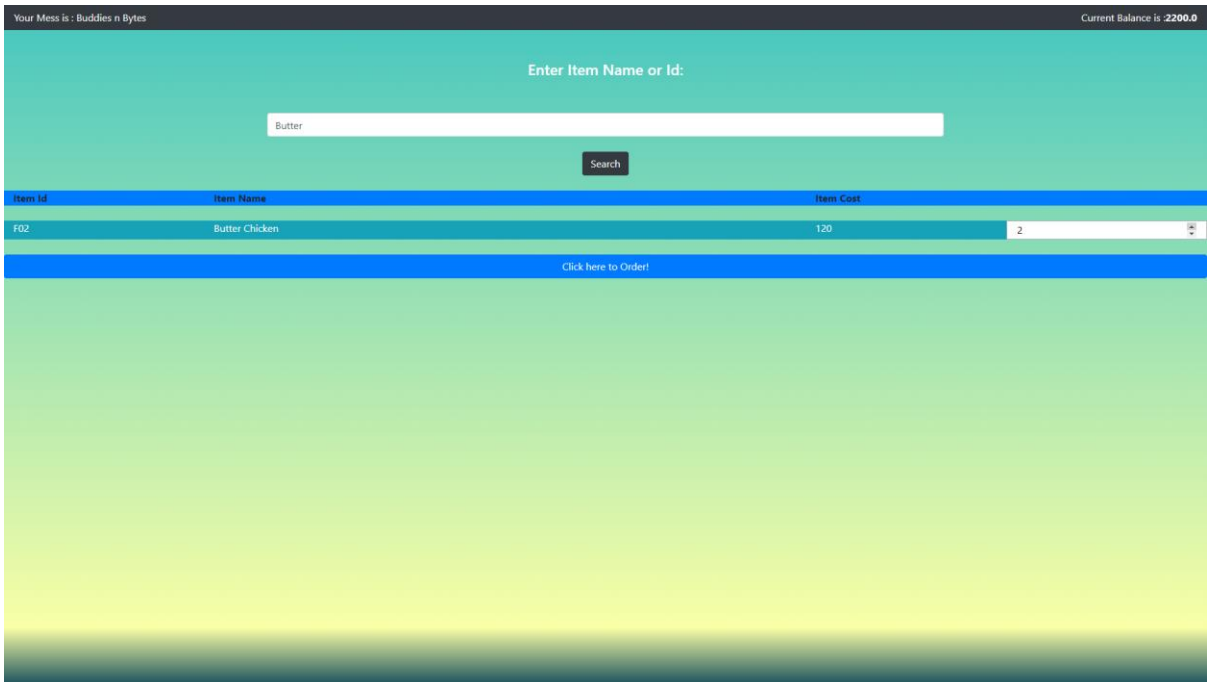
Screen Shots

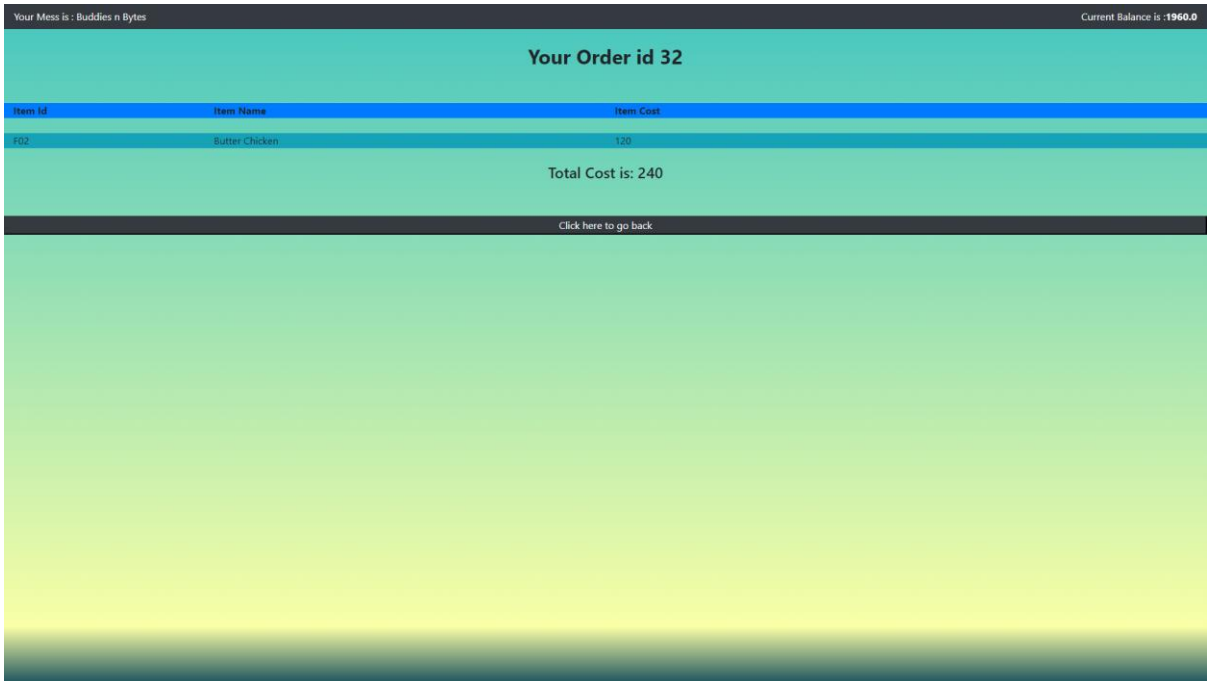
Student Login

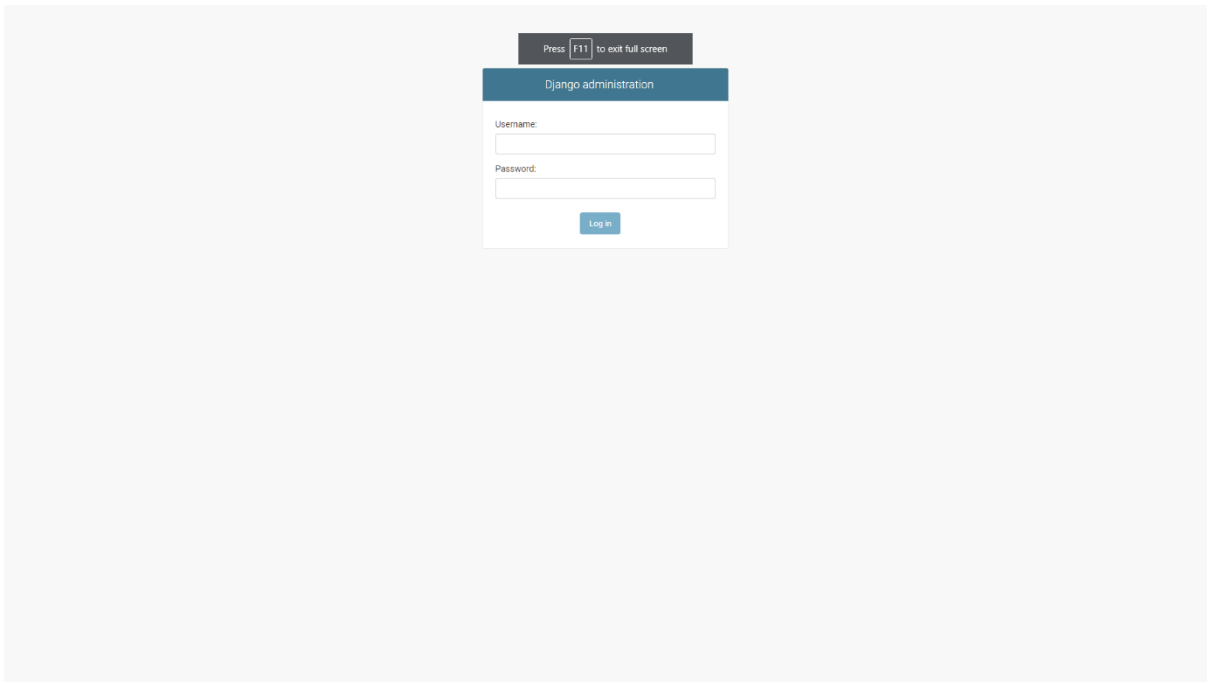


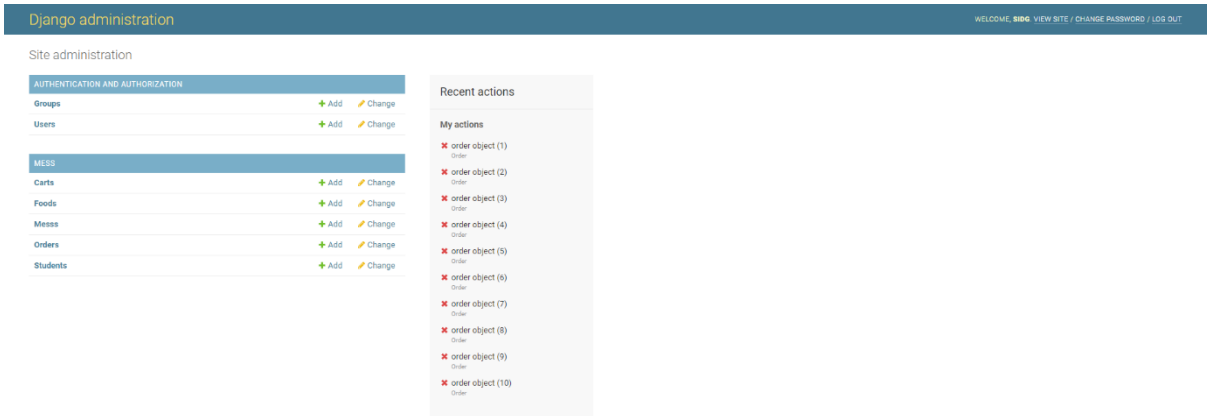
Menu

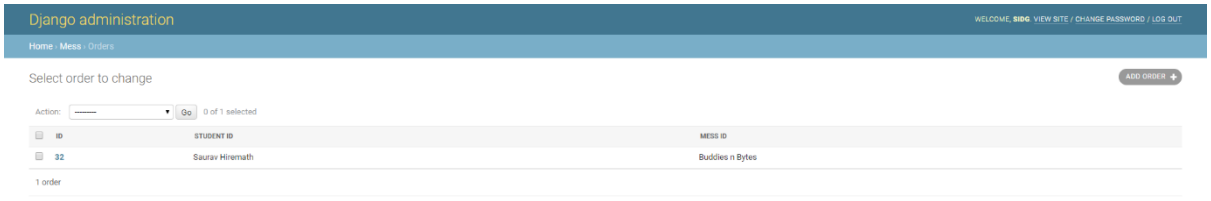

Your Mess is : Buddies n Bytes
Current Balance is :2200.0
Enter Item Name or Id:
Butter
Search
Item Id
Item Name
Item Cost
F02
Butter Chicken
120
0
Click here to Order!

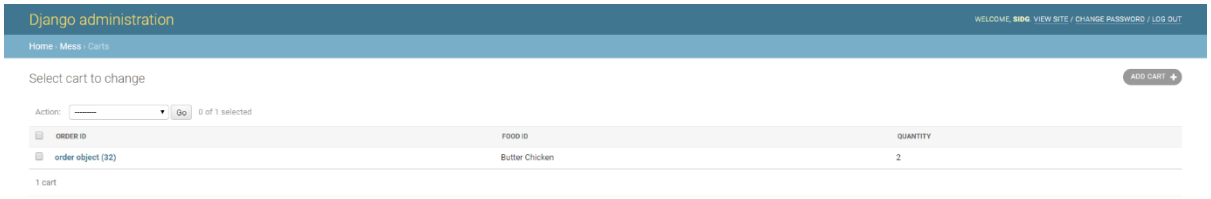

Your Mess is : Buddies n Bytes
Current Balance is :2200.0
Enter Item Name or Id:
Butter
Search
Item Id
Item Name
Item Cost
F02
Butter Chicken
120
2
Click here to Order!


Your Mess is : Buddies n Bytes
Current Balance is :1960.0
Your Order id 32
Item Id
Item Name
Item Cost
F02
Butter Chicken
120
Total Cost is: 240
Click here to go back

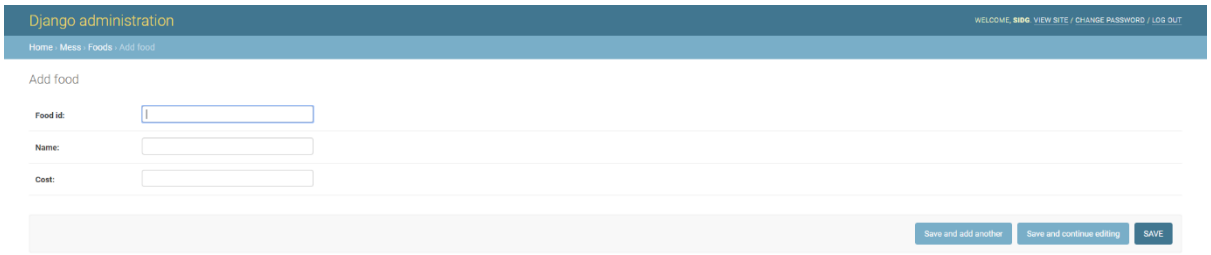

Press F11 to exit full screen
Django administration
Username:
Password:
Log in


Django administration
WELCOME, SIDG VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
AUTHENTICATION AND AUTHORIZATION
Groups + Add Change
Users + Add Change
MESS
Carts + Add Change
Foods + Add Change
Messs + Add Change
Orders + Add Change
Students + Add Change
Recent actions
My actions
order object (1)
order object (2)
order object (3)
order object (4)
order object (5)
order object (6)
order object (7)
order object (8)
order object (9)
order object (10)


Django administration
WELCOME, SIDG VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Mess › Orders
Select order to change
ADD ORDER
Action: ---------- Go 0 of 1 selected
ID
STUDENT ID
MESS ID
32
Saurav Hiremath
Buddies n Bytes
1 order


Django administration
WELCOME, SIDG VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Mess › Carts
Select cart to change
ADD CART
Action: ---------- Go 0 of 1 selected
ORDER ID
FOOD ID
QUANTITY
order object (32)
Butter Chicken
2
1 cart


Django administration
WELCOME, SIDG VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Mess › Foods
Select food to change
ADD FOOD
Action: ---------- Go 0 of 6 selected
FOOD
Daal Makhani
Butter Chicken
Palak Paneer
Tea
Milkshake
Lassi
6 foods


Django administration
WELCOME, SIDG VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Mess › Foods › Add food
Add food
Food id:
Name:
Cost:
Save and add another
Save and continue editing
SAVE

Sample Code

Database Models

```
from django.db import models
from django.contrib.auth.models import User

class student(models.Model):
    Student_id = models.CharField(max_length=100, primary_key=True)
    Name = models.CharField(max_length=100)
    Mess_id = models.ForeignKey('mess', on_delete=models.PROTECT)
    Password = models.CharField(max_length=256)
    Balance = models.FloatField(max_length=100)

    def __str__(self):
        return self.Name

class mess(models.Model):
    blocks=[(chr(i),chr(i)) for i in range(65,82)]
    Mess_id = models.CharField(max_length=50, primary_key=True)
    Block = models.CharField(choices=blocks, max_length=1)
    Name = models.CharField(max_length=50)

    def __str__(self):
        return self.Name
=
```

```
class food(models.Model):
    Food_id= models.CharField(max_length=20, primary_key=True)
    Name= models.CharField(max_length=20)
    Cost= models.CharField(max_length=20)

    def __str__(self):
        return self.Name

class order(models.Model):
    id = models.AutoField(primary_key=True)
    Time= models.DateField()
    Student_id= models.ForeignKey("student", on_delete=models.PROTECT)
    Mess_id = models.ForeignKey("mess", on_delete=models.PROTECT)


class cart(models.Model):
    Food_id= models.ForeignKey("food", on_delete=models.PROTECT)
    Order_id= models.ForeignKey("order", on_delete=models.PROTECT)
    quantity= models.IntegerField()
    def __str__(self):
        return str(self.Order_id)
```

Views.py

```
from django.shortcuts import render
# import logging
# logger = logging.getLogger(__name__)

from django.http import HttpResponse
from django.contrib.auth import authenticate
from django.urls import reverse
from django.views.decorators.csrf import csrf_exempt
from datetime import datetime
from django.db import connection



from .models import mess, food, order, student, User, cart
from .forms import SignInForm

# url = reverse('Home')

def login(request):
    message=""
    if request.method == 'POST':
        form = SignInForm(request.POST)
        foodi=food.objects.all()
        # print(request.POST)
        if form.is_valid():

stud=student.objects.filter(Student_id=request.POST['user_regno'])[0]
            if stud.Password == request.POST['user_pass']:
```

```
            return render(request, "index.html",{'student':stud,'food':foodi})
        else:
            message="Invalid Credentials"

    else:
        form = SignInForm()

    return render(request, "login.html", {"form":form,'message':message})

def home(request):
    try:
        query=request.GET['q']
    except:
        query=''
    if query=='':
        foodi=food.objects.all()
    else:

foodi=food.objects.filter(Food_id__contains=query)|food.objects.filter(Nam
e__contains=query)
    stud=student.objects.filter(Student_id=request.GET['sid'])[0]
    return render(request, "index.html",{'student':stud,'food':foodi})

def orders(request):
    cart1=[]
    fq=[]
    total=0
    print(request.GET.items)
    for i,j in list(request.GET.items()):
```

```
        if i!="sid":
            if int(j)>0:
                f= (food.objects.filter(Food_id=i)[0])
                total+=int(f.Cost)*int(j)
                quan=int(j)
                tot=int(f.Cost)*int(j)
                fi=f.Food_id
                cart1.append(f)
                fq.append([quan,fi])
    stud=student.objects.filter(Student_id=request.GET['sid'])[0]
    Student_id =
student.objects.only('Student_id').get(Student_id=stud.Student_id)
    ords= order(Time=datetime.now(), Student_id=Student_id,
Mess_id=stud.Mess_id)
    ords.save()
    # ords.save()
    # with connection.cursor() as cursor:
    #     cursor.execute("INSERT INTO order values (Time=%s, Student_id=%s,
Mess_id=%s);",[datetime.now(), stud.Student_id, str(stud.Mess_id)] )
    #     row = cursor.fetchone()


    for i,j in fq:
        if(i>0):
            Food_id = food.objects.only('Food_id').get(Food_id=j)
            Order_id = order.objects.only('id').get(id=ords.id)
            crt_item= cart(Order_id=Order_id, Food_id=Food_id, quantity=i)
            crt_item.save()
    stud.Balance= stud.Balance - total
    stud.save()
```

```
    return render(request, 'Cart.html',
{"cart":cart1,"total":total,'qt':fq,'student':stud,'ordid':ords.id})
```

Admin.py

```
from django.contrib import admin


from .models import mess, student, food, order, cart


@admin.register(mess)
class messAdmin(admin.ModelAdmin):
    list_display=['Name', 'Block']
    pass
@admin.register(student)
class studentAdmin(admin.ModelAdmin):
    pass
@admin.register(food)
class foodAdmin(admin.ModelAdmin):
    pass
@admin.register(order)
class orderAdmin(admin.ModelAdmin):
    list_display=['id', 'Student_id','Mess_id']
    pass


@admin.register(cart)
class messAdmin(admin.ModelAdmin):
    list_display=['Order_id', 'Food_id','quantity']
    pass
```